Spring 5-2-2018

Stock Marketing Prediction Using Narx Algorithm

Enas Alkhoshi

STOCK MARKETING PREDICTION USING NARX ALGORITHM

by

ENAS ALKHOSHI

Under the Direction of Saeid Belkasim, PhD

ABSTRACT

Computational technologies have offered faster and efficient solutions to financial sector. In the financial market, the advancements in computational field have been achieved by the use of neural networks and machine learning that delivered a number of financial tools. Thus, in this thesis, we aim to predict the stock index marketing for the “Dow Jones” index by using deep learning algorithms. We propose a model based on an adaptive NARX neural network to predict the closing price of a moderately stable market. In our model, non-linear auto regressive exogenous input model inserts delays into the input as well as the output acting as memory slots thereby raising the accuracy of the prediction. Moreover, Levenberg-Marquardt algorithm has been used for training the network. The accuracy of the model is determined by the mean squared error. We also used LR model, with the same parameters as NARX, to improve the overall accuracy.

INDEX WORDS: Artificial intelligence, stock prediction, NARX algorithm, deep learning, financial forecasting.

STOCK MARKETING PREDUCTION USING NARX ALGORITHM

by

ENAS ALKHOSHI

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of

Master of Science

in the College of Arts and Sciences

Georgia State University

2018

STOCK MARKETING PREDUCTION USING NARX ALGORITHM

by

ENAS ALKHOSHI

Committee Chair: Saeid Belkasim

Committee: Yanqing Zhang

Ying Zhu

Electronic Version Approved:

Office of Graduate Studies

College of Arts and Sciences

Georgia State University

May 2018

ACKNOWLEDGEMENTS

This thesis research is the end of my journey in achieving my Master degree. This thesis has been stayed on track and been seen through to achievement with the support and assistance of numerous people. At the completion of my thesis I would like to acknowledge all these people who made this thesis possible and a wonderful experience for me. At the end of my thesis, it is a pleasant responsibility to express my thankfulness to all those who contributed to the success of this study and made it a wonderful experience for me.

First and foremost, praises and thankfulness to the God, the Almighty, for showers of blessings during my research work to complete the thesis research successfully.

At this moment of accomplishment, I would like to express my deep and sincere gratitude to my research supervisor, Dr. Belkasim Saeid, Associate Professor, Department of Computer Science, Georgia State University, for providing me the opportunity to do research and giving invaluable guidance throughout this research. His dynamism, vision, sincerity, and motivation have deeply inspired me. I am greatly grateful for what he has offered me.

Besides my advisors, I would like to thank the rest of my thesis committee: Dr. Yanqing Zhang, Professor, Department of Computer Science, Georgia State University and Dr. Ying Zhu Associate Professor, Department of Computer Science, Georgia State University, who asked me good questions and gave me valuable comments. Besides this, their encouragement helped me in the successful completion of this work.

I would further like to extend huge, warm thanks to our research group members for their valuable help and support.

I gratefully would like to pay high regards to my Parents for their love, prayers, caring and sacrifices for educating and preparing me for my future. I owe everything to them.

I am very much thankful to my husband Majed Alkhasha and my daughters Malak & Deem for their love, understanding, prayers and continuing support to complete this thesis research.

I am greatly grateful to my brothers and sister for their sincere encouragement and inspiration throughout my thesis research and lifting me uphill this phase of life

Last but not least, my acknowledgments go to all the people who have supported me to complete the thesis research directly or indirectly.

Enas Alkhoshi

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 INTRODUCTION

Machine learning, while useful in most applications, works to demonstrate efficacy in the prediction of stock prices. This chapter investigates the application of the nonlinear autoregressive exogenous model (NARX) architecture to facilitate deep learning for predicting stock prices from economic news. The chapter starts by detailing the nature of financial predictions. It then moves forward to the application of neural networks and artificial intelligence in the discovery of patterns resulting from economic news. The NARX architecture is proposed in this chapter for predicting stock prices. The NARX architecture is known for its versatility and ability to provide efficient prediction of stock prices.

Financial markets are keen on stock price prediction that could determine the future value of a stock thus minimizing losses. The financial market is a volatile environment. Therefore, having a means of achieving precise future predictions is desirable for purposes of realizing a profit. The prediction of stock prices is of imperative importance for purposes of making buying or selling decisions [22]. However, stock predictions have in the past been disqualified by economists who believe in the Efficient Market Hypothesis (EMH). The hypothesis postulates that it is impossible to forecast the value of stocks. However, researchers in the field of stock index predictions have reported considerable success, thus debunking EMH [22]. As a consequence, a significant number of researchers have developed several stock prediction techniques.

Additionally, the stock market has been quantified as "informationally efficient" [13]. The theory of informationally efficient markets postulates that new information about a company is known with certainty, and it immediately contributes the value of the company or to the stock of the company [13]. Therefore, unlike in EMH which disqualifies stock prediction, the informationally efficient theory only concerns itself with new news. Therefore, predictors no longer have to dwell on the ominous task of accounting for entire space of information available to make predictions. Rather, they can focus on new news related to a particular company to make predictions of that company's stock movement. Therefore, the informationally efficient theory has propelled interest in stock index predictions. In relation to the stock index, or the stock market index, the information efficiency theory makes it plausible for researchers to accurately predict the value of a section of the stock market. Therefore, as opposed to predicting the movement of a single stock, researchers can apply predictive mechanisms to compute the prices of selected stocks from news related to the stock industry as well as emerging news about the selected companies.

The advent of capable new computer algorithms has also fueled the race to improve stock index prediction. The recent progress in machine learning algorithms, has made predicting the movement in stock prices an achievable task instead of a daunting one. Manual aggregation of new information about a company and using the information to make stock index predictions is impossible in today's fast evolving stock markets. Machine learning techniques, on the other

hand, can be used to uncover patterns in the movement of the stock market index in relation to any related news. These patterns help a computer system to dynamically discover new patterns by associating them with previously learned ones. Thus, computers can learn from previous data to make predictions based on the existing data.

The NARX algorithm is one such use case example where computers, through past data, can be trained to make accurate predictions in stock index movements. Through time series modeling, the nonlinear autoregressive exogenous model (NARX) can accommodate both past values of the same series as well as current values. Financial forecasting founded on time series data has, over the last several years, been an area of interest for researchers due to its potential accurate predictions [4]. NARX can be successfully deployed to study patterns in information and the resulting changes in the stock index market. The algorithm also has an error term that that can be used to optimize the current value of the time series to improve predicted precision

1.1 Financial Forecasting

Financial forecasting deals with the financial performance of a company by leveraging historical data to determine the future financial performance of the company. As such, researchers can use financial forecasting to uncover events that affect the financial performance of a company, and thus the performance of its stock price. Data such as sales expectations can decrease or increase the price of shares, and therefore they can be used to plot the stock index

movement for a selected group of companies. Additionally, analysts can look into the revenues of selected companies and compare them to economic indicators to determine the stock index movement. The variables chosen in the conduct of financial forecasting are often based on the passage of time in relation to the occurrence of specific events.

The history of financial predictions dates back to a study conducted by Fama in 1970 on the efficiency of the market hypothesis [13]. Fama postulated that it was impossible to achieve consistent prediction on stock prices since market prices already incorporated and reflected all available information. However, this hypothesis was problematic. Firstly, it is difficult to quantify that market prices captured all available information. Secondly, not all of the available information is of use to investors in the market. Investors react to the market at various times, looking at different information, and arriving at different conclusions at different times. Fama appropriately confirmed that markets are driven by information. However, the researcher failed to factor the behavior of investors. Investors take on different tactics and strategies in the financial market. Some are risk-averse while others are exogenous traders. Different groups of investors are capable of overreacting, underreacting, or normal reactions on a given set of market information. Additionally, it is impossible to quantify the impact of economic news on the stock prices. Therefore, belief in EMH means that investors will never make money in the market. That is a misnomer, especially given the fact that investors continually make money on the stock market [26].

Believing in EMH also means that a person also believes in the random walk hypothesis (RWH) as postulated by Louis Bachelier, The Theory of Speculation suggests that stock price profitability followed a random walk. The square of a typical amplitude of return fluctuations increases in proportion to time. Therefore, if this proportional relationship does not exist, then the price changes are completely random. An attempt at proving this theory revealed that stock returns are nonrandom, and consequently, it is possible to carry out stock forecasting [26].

Thusly, the movement of stock prices is a reaction to the trend in investors' hopes, fears, knowledge, optimism, and greed. The totality of these feelings are perceivable in the stock price level, which in turn is not the actual value of the stock, but a reflection of what people perceive them to be worth. As a consequence, the financial prediction is possible through the observance of economic news, how people historically react to the news, and how the expected reaction affects the stock prices. Therefore, it is possible to teach a computer algorithm, using a large data set, to observe patterns in economic news reaction, and how the reaction affects stock prices [2]. The model constructed under the auspices of machine learning can then be applied to current news to predict stock price movements in relation to the contemporary news. This hypothesis was formulated by Graham and Dodd in their book, *Security Analysis* [2], where the authors qualified the assumption that market activities are driven by prices from the consumers and suppliers. In finance, stock prices are then driven by the price, quantity of stock, and the time factor.

Stock market predictions and financial forecasting in general can be achieved in two methods [28]. One method, the fundamental analysis (FA) is concerned with the company as opposed to the stock price and volume data. Analysts using the FA method to pick a stock based on the performance of the company. The second method, technical analysis (TA) is limited to the price data. Machine learning efforts predominantly use the TA method to make financial predictions in the stock index market.

1.2 Neural Network in Stock Index Prediction

Researchers have successfully used news analytics and market sentiments to successfully predict stock prices [5]. Additionally, researchers have been able to leverage online social networks (OSNs) to gain an understanding of market sentiments. These two data sources have aggregated a solid source of information for purposes of predicting stock performance.

Artificial neural networks are comprised of nodes, similar to those on a vast network of neurons in the human brain. They are a mathematical representation of the human brain [5]. The artificial neuron network is constructed such that output from one neuron acts as input into another artificial neuron. The input layer in the artificial neuron network is made up of recorded values that are supplied to the next layer of neurons. The hidden layer in the network receives the input and manipulates it for purposes of constructing a prediction. The output layer consists of nodes assigned to each class. Therefore, a single step through the network results in the

assignment of value to each output mode. The result is assigned to the node class with the highest value.

As such, the system can learn and progressively improve its performance by studying examples. However, in an artificial neuron network, the system does not require to have any prior knowledge of the task it is performing. The network processes one record at a time and learns by comparing the prediction of the record to an actual record. If the system makes erroneous initial predictions, these are used as input into the system for purposes of modifying the underlying algorithm for the second iteration of the prediction. These iterations are performed numerous times until the network can make accurate predictions [5].

During the supervised training phase of an artificial neural network, the correct class for each record is known in advance. The output nodes are then assigned the correct value, “1”, for each node corresponding to the correct class. Other nodes are assigned a “0” value to indicate they are false. The results are realized using the values of 0.9 for correct values and 0.1 for false values. Therefore, it becomes possible to calculate the correct values for the output nodes and calculate the error value for each node. The error terms are then used to adjust the algorithm in the hidden layer such that in the next iteration, the output values will be closer to the correct value. The iterative learning steps are of imperative importance in a neural network. During these steps, the records are presented to the network individually and the weights (error terms) corresponding to the input values are adjusted during each iteration. Once every class has been

presented, the process is repeated. In every iteration, the neural network trains itself by using the error terms to adjust its algorithm to predict the correct class label for the provided input training data set [5]. This process makes neural network predictions consistent and tolerant of noisy data. The network is also able to classify patterns which it has not been trained on. Therefore, neural network predictions are suitable for the prediction of stock price movements irrespective of the information supplied as the input.

For neural networks to be effective in predicting the desired results, it is essential to provide the correct number of layers as well as the processing elements for each layer. The rule of the thumb is to increase the number of processing elements in the hidden layer as the complexity in the relationship between input data and the desired output increases.

1.3 Deep Learning

Deep learning, a branch of machine learning, attempts to make high-level abstractions in data by deploying multiple neural layers. The primary goal of deep learning is to move machine learning towards its original goal of achieving artificial intelligence. In deep learning, unlike machine learning, the primal focus is on data representation. Machine learning, on the other hand, focuses on optimizing takes-specific algorithms.

Deep learning, therefore, makes the use of large neural networks trained with large data sets compared to the ones used in machine learning. Therefore, unlike machine learning efforts

that often reach a plateau even when new data sets are used, deep learning continues to increase its performance.

As such, deep learning allows neural networks to learn representations of data with multiple levels of abstraction. Deep learning application on neural networks allows for the discovery of intricate patterns and structures in large sets of data through the use of back propagation algorithm [24], which define how the machine should change its internal parameters used to compute the representation in each layer as contrasted with the representation in previous layers.

The back propagation algorithm is at the centre of deep learning. It refers to the backward propagation of errors and it is generally used in supervised learning of artificial neural networks through gradient descent. The algorithm takes input from the artificial neural network and its error function to calculate the gradient of the error function in respect to the average weights computed by the network.

The primary advantages of deep learning are therefore apparent. It is highly accurate compared to ordinary neural networks. Also, it is highly versatile compared to the traditional implementation of artificial neural networks. However, it requires considerable computing power as well as being time-consuming during the network training stage.

1.4 Deep Learning Approaches in Predicting Stock Prices

Since artificial neural network approaches have resulted in less than impressive results in predicting stock prices due to the high noise levels in stock data. Therefore, deep learning approaches are preferable [28]. Deep learning approaches are favored owing to the fact that they have successfully been applied in image classification, Natural language processing, and speech recognition tasks. Thus, the deep learning approaches can be used in the processing of stock data since they are also time series [28].

One of the deep learning approaches in predicting stock data is based on 2-Directional 2-Dimensional Principle Component Analysis ($(2D)^2PCA$) [21]. In this approach, $(2D)^2PCA$ is used for dimensionality reduction followed by deep neural network to predict stock prices. The particular deep neural network used in this approach is a multi-layer feed forward neural network that employs supervised learning. In the output layer, which has two nodes, the network employs the softmax function for classification and a linear function for regression. The network learns when weights are adapted to minimize the error on the training data. For purposes of updating the weights and biases within the network, the Stochastic Gradient Descent (SGD) supervised training algorithm is used. However, the application of this approach did not accurately predict Total Return and RMSE compared to other algorithms.

Another deep learning stock price prediction approach is the recurrent Convolutional Neural Networks [15]. The convolutional layer in this neural network performs one-dimensional convolution (temporal convolution) capturing information from sentences contained in economic

news articles. Secondly, the system performs temporal max-pooling on the one-dimensional output from the convolution for purposes of capturing the most important information. The system applies a rectifier linear unit for purposes of introducing a non-linearity into the model. However, it is worth noting that the high quantity of parameters makes recurrent Convolution Neural Networks susceptible to over fitting. Therefore, it is imperative to use regularization techniques [15].

Several layers can be used are current layers of the recurrent Convolution Neural Network. One of the layers that follow the convolution layer is used for interpreting the output in the desired iterations. Long Short-Term memory recurrent neural network architecture has been used successfully to introduce a new memory cells in the recurrent layer [21]. The architecture constitutes several gates for decision making. The forget gate in the architecture makes a decision on which information in a memory cell is worth forgetting. The input gate decides on the values in a memory that need updating with an input signal, while the output gate makes the decision of whether a memory cell should affect other neurons or not. The advantage of this structure is that it allows modeling of long-term dependences thereby preventing the vanishing gradient problem present in other neural network architectures. Although this model achieves consistent and accurate results by analyzing news from the previous day, it does not outperform the event embedding convolution neural network. Therefore, the recurrent convolution neural network would provide biased predictions as a result of this.

The NARX algorithm is another implementation of deep learning techniques in the analysis of stock prices. NARX is a time series model that can be trained with different algorithms. It relies on a feed forward neural network architecture that regresses the upcoming value of the output with the previous output as well as the previous value of the independent exogenous input signal. The NARX algorithm is able to predict the next value of the input signal due to its implementation. Additionally, the algorithm is capable of being deployed as a non-linear filter for purposes of removing noise from the output, unlike other deep learning prediction implementations. Since the output of a NARX network is estimated and provided as feedback to the input, NARX algorithms are capable of multi-step predictions. They are therefore faster compared to other prediction algorithms, and as such favorable for predicting stock prices from large data sets of information.

Another advantage of NARX is its ability to represent various nonlinear dynamic behavioral patterns such as the ones observed in the stock market. The algorithm can make these predictions despite the apparent high level of distortion in the provided data, as long as the training set adequately contributed to the training of the underlying neural networks. It can reconstruct events in the stock market, and therefore make reliable future predictions. The simulation provided by NARX shows promise in the adoption of the weighted cost function in advancing the reliability of peak function.

1.5 The NARX Algorithm

Nonlinear Autoregressive model with exogenous input (NARX) is a recurrent dynamic network that has feedback connections engaged in several neuron layers [17]. The architecture of NARX is based on the ArX model predominantly deployed in time-series modeling. ARX (Autoregressive exogenous) models take on exogenous inputs such that they take up current values of time series as well as past values of the same time series. As such, NARX has powerful classes that fit well in the modeling of nonlinear systems and time series. Research shows that NARX demonstrates superior supervised learning and therefore, it is more effective compared to other neural network architectures. The NARX architecture is also known to converge quicker and generalize better compared to other neural networks.

The equation that summarizes the function of the NARX architecture is

$$y(t) = F\,(y\,(t\text{-}1), y(t\text{-}2), y(t\text{-}n_Y), u(t\text{-}1), u(t\text{-}2), u(t\text{-}n_u) \qquad (1)$$

The next value of the dependent output is $y(t)$ and it is regressed on preceding values of the output as well as the preceding values of the independent (exogenous) input $u(t)$. The NARX architecture is commonly implemented in a feed-forward neural network on the appropriate function (F).

Apart from the above equation, another important aspect of the NARX architecture, especially during training, is that the output can be estimated to be the output of a nonlinear dynamic system that is being modeled. In this case, it can be the nonlinear dynamic output of the

stock market price movement. Since the real output is already known during training, the implementer can construct a parallel architecture where the true output is used in the feed-forward as opposed to the estimated output like common in most deep learning implementations. This approach gives NARX several advantages. One of these advantages is that the architecture's gains higher accuracy in predictions. Another advantage is that the resultant neural network adopts a pure feed-forward architecture where static backpropagation can be deployed for training purposes since the network is purely feedforward without any feedback.

The NARX system, predictive time series tool, works by having its input layer, hidden layer, delay and activation functions declared to attain the anticipated results. The activation of the functions is achieved through Bayesian Regularization, Lavenberg-Marquardt, and Scaled Conjugate Gradient. Bayesian Regularization provides a network training task that brings the weights and bias values up to date in relation to the network. It diminishes the squared error value in the network by accurately selecting the weights and bias. The Lavenberg-Marquardt training offers a superior training algorithm. From these functions, NARX can determine a framework for predicting the desired values, such as the stock price, depending on the input and output data [1]. Increasing the number of neuron layers' results in generating fewer errors in less iterations. However, additional layers of neural networks do not necessarily result in less error. Instead, the implementer of the NARX network should take note of relation between the number of layers and the number of iterations that result in fewer errors. Such efforts will guarantee a

network that is capable of generating desirable predictions. The network will be capable of making these predictions in the minimum amount of time, regardless of the data set presented to the network for analysis.

2 THE LITERATURE REVIEW

The state of the stock market has over the past number of years has become significantly important as the number of stocks as well as investors has reached high records. Forecasting of the returns on the stock market using software packages gained considerable attention by many investors and financial institutions. Accurate stock price forecasting is very critical as it directly impacts the decision of investors and their plans to invest. The stock market is a rather non-linear deterministic system that appears significantly random primarily due to its irregular nature and fluctuations. In order to develop an accurate system for predicting stock prices, it is necessary to understand the nature of the stock market and the behavior of investors. According to Rajput &Kaulwar (2017), researchers have proposed various fundamental, technical as well as analytical methods of stock price prediction [8]. However, none of these techniques have established an effective approach of predicting the nature of the stock market.

The increased need to accurately predict the prices and the behavior of stock exchanges has led to the development and emergence of a large number of stock prediction approaches and techniques. These techniques vary greatly and considerably depending on the availability of information, the quality, accuracy and correctness of modeling as well as the assumptions

applied in the design of the model. The dependence of the modern financial sector on automating many operations including stock price prediction is the main driving factor for the recent advances in financial software development. Primarily, the recent advances in neural networks and machine learning has established efficient tools suitable for use in the financial market [17]. In the review that follows, we assess and comprehensively analyze the existing works of literature on the application and the use of NARX deep learning, a neural network-based technology, in the prediction of stock index.

According to the article by Chaigusin, Chirathamjaree & Clayden (2008), neural networks have been regarded as the most suitable and appropriate tool for the prediction of stock exchange behavior. Unlike other methods and techniques that are primarily based on discrete functional forms of data representation and processing, neural networks enable for the ability of the systems to learn patterns as well as relationships present in the data. An aspect worth noting in the stock industry is that prediction of the stock prices is rather a complex and challenging operation. As a result of these complexities and difficulties, most approaches and methods that have been developed are ineffective.

In the past decade, a growing number of researchers and scholars have invested both time and money in an attempt to develop and enhance approaches based on artificial neural networks that can handle and support successful stock market price prediction. Cavalcante, Brasileiro, Souza, Nobrega& Oliveira (2016) for instance presented a review of the modern and recent

computational intelligence techniques that have been designed to solve financial problems and stock challenges. The paper primarily focused on the application of machine learning on the stock prediction. Another study by Al-Shayea (2017) suggested the application of neural network approach for the technical and theoretical analysis of the financial market, as well as its application in the timing of buying and selling of stock. In their research, a learning method is proposed towards the improvement of the accuracy of the prediction, using the already available information.

Numerous models have been designed by scholars and designers for the realization of of neural networks in process of prediction of stock values. Figure 1 shows the main components of an existing stock prediction system that uses a neural network [17]. The model makes use of the feed forward multi-layered perceptron employing back propagation model to train the neural network. The main problem with this model is that, due to its nature, it fails to adequately and effectively account for the volatility and the continuous fluctuations of the stock prices as such significantly reducing the accuracy of its predictions [17].

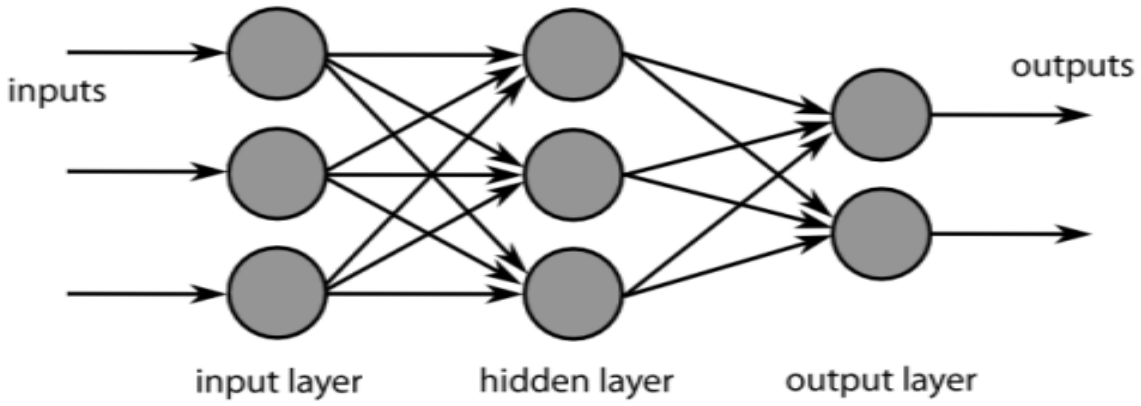


Figure 1: Traditional Neural Network Model for Stock Prediction

In comparison to the traditional model depicted above, the NARX model takes a considerably different format allowing feedback loops in the architecture. As shown in the figure below, the implementation of an NARX stock prediction neural network allows for the input and the output to take a multi-dimensional form [1]. There are many different applications of the NARX neural network prediction system. First and foremost, the model can effectively be used to predict the next possible value based on some inputs. Secondly, it can effectively be used for non-linear filtrations. In this case, the system is ideal for producing noise-free outputs [1].

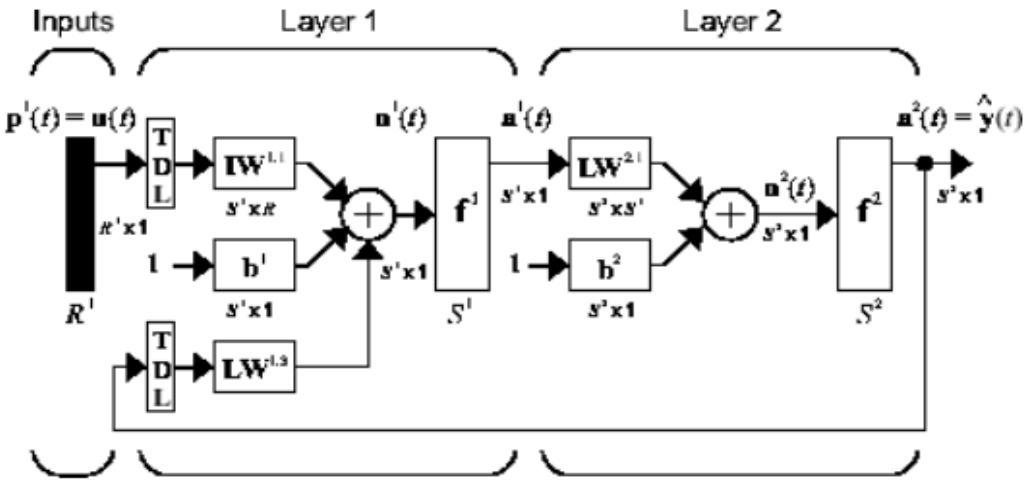


Figure 2: NARX-based stock prediction model

A considerable innovation of NARX neural networks has been produced by Diaconescu (2008). Diaconescu (2008) clearly proved and showed that an architectural approach of the neural networks with an embedded memory employingthe Nonlinear Autoregressive model with an exogenous input (NARX) is very effective for the prediction of the stock behavior. According to his observation, the performance and accuracy of the NARX neural networks is attributed to the chaotic fractal time series that is often applied as inputs for the neural network. The study conducted by Wang (2015) also depicted similar results showing the effectiveness of NARX neural network systems in the prediction of stock prices.

The realization of the benefits and the applicability of NARX in financial markets has been primarily propagated by its applicability in other non-financial industries. Thakur, Tiwari, Kumar, Jain & Singh (2016) for instance conducted an assessment into the use of the NARX neural networks for the prediction of petrol prices. The petroleum industry has many

characteristics and attributes that it shares with the financial market. As explained in Thakur et al. (2016), the past number of years have particularly been witnessed considerable fluctuations in the prices of crude oil, with the prices increasing and then significantly declining in an irregular manner. In their analysis, Thakur et al. (2016) found out that NARX neural networks could efficiently and effectively be applied for the accurate prediction of crude-oil prices. The research observed that the mean square error and the prediction errors of the network declined with increased training of the neural network.

The hypothesis that NARX neural networks can effectively be applied in the financial market have been followed by a series of research work assessing the practical application of these systems in the stock market in different regions and countries in the world. The work by Ercan (2017) focused on assessing the application of the NARX systems in the prediction of the Baltic stock market. According to the article, artificial neural networks are not commonly utilized in the financial market in most Baltic nations. The research made use of the index value and the EUR/USD towards the prediction of the index. Input data was collected for a period of four years from 1st January 2013 to 1st January 2017. The data was then divided into two sets, one comprising of the data that will be used for the purposes of neural network training and the other comprising of data to be used in testing the neural network. According to the results of the research, the Baltic stock market values could correctly be predicted by the use of a NARX neural network [20].

Prediction based on neural networks has been observed to be considerably suitable for forecasting in non-linear time series applications. Of particular importance is that with the use of NARX, it is not necessary to have information on the cause of signal [6]. As compared to other forms of neural networks, NARX is a recurrent and a dynamic network, with looping feedback connections that enclose the different layers of the neural network.

The ARX model is particularly used in the prediction and forecasting of the financial information and in other forms of time-series modeling. There are a number of reasons and benefits why the model is considered the most appropriate for use in the prediction of the stock market. According to Shahbazi, Memarzadeh & Gryz (2016), this class of models have specifically been observed and determined to be well suited in modeling non-linear systems such as the stock market. It has also been demonstrated that learning in NARX is more effective as compared to learning in other forms of neural networks. The NARX has also been observed to converge much faster and to generalize results and to draw conclusions in a more effective and better way than other forms of neural networks [6].

While the stock market prediction remains to be a major challenging issue in the development of research and technology, several attempts have been successful in presenting reasonably accurate systems. Moreover, although stock market prediction has already been implemented using neural network approaches, the NARX model presents a considerably effective and accurate approach. Particularly, the ability of the NARX model to incorporate time

delays results in an improved accuracy in the computation of the stock values based on the current as well as the past information. However, even though the NARX model enables for accurate and simple computation of the stock prices, it does not consider a number of external factors such as the political influence or natural disasters. As such, previous research has clearly identified the need for the NARX to incorporate the impact of external factors in the prediction of the stock prices.

3 STOCK PREDICTION METHODOLOGY

To illustrate our approach, we are going to highlight the subjects that are comprehensively discussed, which include.

METHODOLOGY:

3.1 Problem Definition

Stock Market Prediction is the most critical goal for investors since its existence. Every day billions of money are traded on the exchange, and behind each dollar is an investor willing to profit in one approach or another. Whole businesses may get increase or decrease in value every day based on their performance in the stock market. Moreover, predicting market changes correctly promises wealth and influence for investors. As result, the prediction of stock index and its associated difficulties pave the way to avoid crisis and help save money [14].

After highlighting the background of the influence of neural networks and deep learning in the finance as well as emphasizing the importance of using NARX to predict the closing price; we will formulate the problem as follows.

$$t(i-n), t(i-n-1), t(i-n-2), \dots\dots\dots, t(i) \approx r(i-n), r(i-n-1), r(i-n-2), \dots\dots, r(i) \qquad (2)$$

i = (the day), t= (target value), r= (return value).

Where n=1,2, 3,…,

i=1,2, 3,…

The problem can be stated as:

We need to find the closing price of a particular stock on the (i+1)th day based on the Stock Close Price on the ith day.

The goal of this work is to predict whether future daily closing price of Dow Jones Industrial Average (DJIA) can influence the prices of stocks. Thus the problem can be treated as a Multi-Class Classification.

The majority of the dynamic systems have been engaged systems, with the flow just at the information layer, or feedforward systems. The nonlinear autoregressive system with exogenous

information sources (NARX) is an intermittent dynamic system, with input associations encasing a few layers of the system. The NARX display depends on the straight ARX show, which is usually utilized as a part of time-arrangement [17]. The defining equation for the NARX model is

$$y(t) = F(y(t-1), \ldots\ldots, y(t-2), y(t-n_Y), u(t-1), u(t-2), \ldots\ldots, u(t-n_u)) \quad (3)$$

To clarify, the new value of the dependent output signal y(t) is based on past values of the output signal; $y(t-n_Y)$ and past values of an independent (exogenous) input signal; $u(t-n_u)$.

So, the basic structure of a NARX network can be modeled as below:

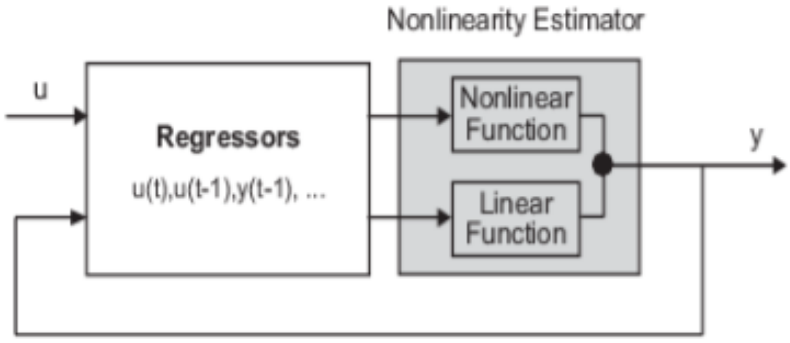


Figure 3:The basic structure of a NARX network

3.2 The Proposed Solution

Our solution's framework, is based on estimating the closing price of a particular stock by employing deep learning methods, we plan to use all relevant information that can be extracted from Dow Jones Industrial Average (DJIA) as an input for our proposed system. In particular,

we plan to use NARX with more than six inputs and the closing price as a target value to create a model of decision-making under uncertainty.

In this model, we calculate the root mean square error between the predicted value and the target value.

The NARX neural network is applied to foresee the time-series data. Moreover, there are different ways to deal with the accomplishment of the estimation of the function for NARX demonstrate. The model in our study is a Feed-forward Neural Network including an input layer with a hidden layer with ten neurons.

$$y(n+1) = F_{NARX\left(\begin{matrix}(x(n),x(n-1),\ldots\ldots,x(n-d_x+1);\\(y(n),y(n-1),\ldots\ldots,y(n-d_y+1)\end{matrix}\right)} \quad (4)$$

NARX is a main class of discrete nonlinear system,

where (·) is a nonlinear function, which can be expressed as followed:

Where $x\,(\cdot) \in \mathcal{R}$, $y\,(\cdot) \in \mathcal{R}$, represent the input and output of the model in discrete-time formula, so as to maintain its generalization, the dead-time parameters is set to 0. Hence, the input vector is defined as $[x(n); y(n)]$ [23].

The neural network is demonstrated in Figure (4),

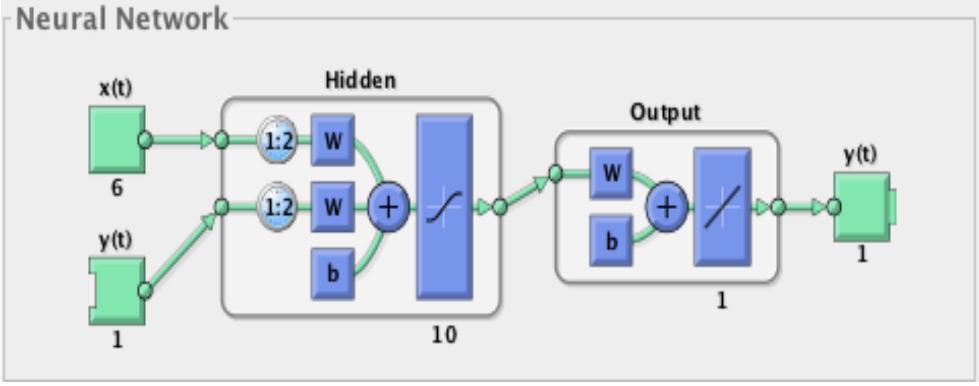


Figure 4: The proposed model of a NARX network

This section illustrates the algorithms we developed for our proposed solution. The model uses a time series method to analyze and predict the closing price. The NARX model is trained using more than six input values - the opening price of the stock, the highest price of the stock, the lowest price of the stock, the volume of the price of the stock, and two more features for the input values for the day. On another hand, the target values are also fed to the network as it is a supervised learning model.

The NARX is implemented utilizing a feedforward neural system to estimate the function f. where a two-layer feedforward network is used for the estimation. This execution likewise takes into account a vector NARX display, where the information of input and output can be multidimensional. The essential MATLAB functions of a neural network are listed in Table (1):

Functions:

Function	Description
nnstart	Neural network getting started GUI
view	View neural network
timedelaynet	Time delay neural network
narxnet	Nonlinear autoregressive neural network with external input
narnet	Nonlinear autoregressive neural network
layrecnet	Layer recurrent neural network
distdelaynet	Distributed delay network
train	Train neural network
gensim	Generate Simulink block for neural network simulation
adddelay	Add delay to neural network response
removedelay	Remove delay to neural network's response

closeloop	Convert neural network open-loop feedback to closed loop
openloop	Convert neural network closed-loop feedback to open loop
ploterrhist	Plot error histogram
plotinerrcorr	Plot input to error time-series cross-correlation
plotregression	Plot linear regression
plotresponse	Plot dynamic network time series response
ploterrcorr	Plot autocorrelation of error time series
genFunction	Generate MATLAB function for simulating neural network

Table 1: The MATLAB 's functions of a neural network

Besides, Levenberg-Marquardt algorithm has been utilized for training the network. The mean squared error is used to calculate the accuracy of the network. Thus, the usage of a open loop l for creating predictions and the accuracy of each case is determined to analyze the effectiveness of a NARX neural network and to determine the best configuration [18].

4 THE STOCK PRICE PREDICTION ALGORITHM

This section illustrates the Stock Price Prediction algorithm we employed in our solution.

Algorithm 1. Nonlinear Auto Regressive eXogenous Neural Network with delays Approach.

Input: Time series data for the opening, low, high, volume, DEXUSUK, and DEXUSEU stock index vector

Output: Time Series data for predicted closing stock index vector

---

4.1 preparing the data

4.1.1 *First, divide the data to input and target value.*

we need to make a pair of input and target vectors for our network. So, in order to divide our data correctly we need to make sure than an input value x will provide a desired output value y.

In this case we let the network learn from such "x" input to predict that "y" value which is already the known target. "x" and "y" should have the same length but not necessarily the same size.

we used 6 input (Open, High, Low, Volume, DEXUSUK, and DEXUSEU), and the target value is the Closing price.

4.1.2 Second, Data Normalization.

Normalization of data in general means the process of preparing and making the data stable for processing. Moreover, in our case Normalization is simply a scaling procedure. Normalization is one of the main parts of ANN learning process because real data obtained from experiments and analysis most of the time are distant from each other. The effect is great because the common activation functions such as sigmoid, hyperbolic tangent and Gaussian, produce results that ranges between [0,1] or [-1,1]. Normalization is used to make data statistically comparable, therefore, it is always beneficial t to normalize the data prior to feeding it to a neural network. The common normalization approaches include Statistical normalization (using mean and standard deviation) and Min-Max Normalization.

4.1.3 Third, Convert data to neural network suitable format.

We preprocess the data to be fed to the defined model. The preprocessing stage of the data is necessary for assuring a successful run under the NARX neural network as well as satisfying the network requirements and meeting its format.

4.1.4 Forth, Divide data into Train Test and validation splits.

In training multilayer networks, the general practice is first to partition the dataset into three subsets. The first part of the dataset is the training set. The training is used for computing the gradient and updating the network weights and biases. The second part of the subset is the validation set. In the validation set, the error is checked through the training process. Also, the validation error typically declines through the early phase of training, equally does the training set error. However, the error on the validation set usually starts to increase, when the network starts to over fit the data. So, the weights and biases of the trained network are saved at the minimum of the validation set error. The essential MATLAB functions for dividing dataset for a neural network is listed in this Table.2 below:

Function	Algorithm
dividerand	Divide the data randomly (default)
divideblock	Divide the data into contiguous blocks
divideint	Divide the data using an interleaved selection
divideind	Divide the data by index

Table 2: The MATLAB functions for dividing data for a neural network

4.2 Data Training

In the training stage, the correct class for each record is known (this is termed supervised training), and the output nodes can, therefore, be assigned "correct" values -- "1" for the node

corresponding to the correct class, and "0" for the others. (In practice it has been found that using values of 0.9 and 0.1, respectively yields better results. It is thus possible to compare the network's calculated values for the output nodes to these "Target" values and calculate an error term for each node. These error terms are then used to adjust the weights in the hidden layers so that, hopefully, the next time around the output values will be closer to the " Target" values [25].

A main feature of neural networks is an iterative learning process in which data cases (rows) are presented to the network one at a time, and the weights associated with the input values are adjusted each time. After all, cases are presented, the process often starts over again. During this learning phase, the network learns by adjusting the weights to be able to predict the correct class label of input samples.

The network processes the records in the training data one at a time, using the weights and functions in the hidden layers, then compares the resulting outputs against the desired outputs. Errors are then propagated back through the system, causing the system to adjust the weights for application to the next record to be processed. This process occurs over and over as the weights are continually tweaked. During the training of a network, the same set of data is processed many times as the connection weights are continuously refined [25]. MATLAB training algorithms are shown in Table.3

Training Algorithm	Training function	Parameters
Levenberg-Marquardt	trainlm	mu, mu_dec, mu_inc, mu_max, epochs, show, goal, time, max_fail, min_grad, mu, mu_dec, mu_inc, mu_max and mem_reduc.
BFGS Quasi-Newton	trainbfg	epochs, show, goal, time, min_grad, srchFcn, scal_tol, alpha, beta, delta, gama, max_fail, low_lim, up_lim, minstep, maxstep and bmax.
Resilient Back propagation	trainrp	epochs, show, goal, time, min_grad, delta0, max_fail, delt_inc, delt_dec and deltamax.
Scaled Conjugate Gradient	trainscg	epochs, show, goal, time, min_grad, sigma, max_fail and lambda.
Conjugate Gradient with Powell/Beale Restarts	traincgb	epochs, show, goal, time, min_grad, srchFcn, max_fail, scal_tol, gama, alpha, beta, delta, up_lim, low_lim, maxstep, minstep and bmax.
Fletcher-Powell Conjugate Gradient	traincgf	epochs, show, goal, time, min_grad, srchFcn, max_fail, scal_tol, gama, alpha, beta, delta, up_lim, low_lim, maxstep, minstep and bmax.
Polak-Ribiére Conjugate Gradient	traincgp	epochs, show, goal, time, min_grad, srchFcn, max_fail, scal_tol, gama, alpha, beta, delta, up_lim, low_lim, maxstep, minstep, and bmax.
One Step Secant	trainoss	epochs, show, goal, time, min_grad, srchFcn, max_fail, scal_tol, gama, alpha, beta, delta, up_lim, low_lim, maxstep, minstep, and bmax.
Gradient Descent	traingd	epochs, show, goal, time, max_fail, min_grad and lr.
Gradient Descent with Adaptive Learning Rate	traingda	epochs, show, goal, time, min_grad, lr, mc, max_perf_inc, lr_dec, lr_inc and max_fail.
Gradient Descent with Momentum	traingdm	epochs, show, goal, time, max_fail, min_grad, lr and mc.
Variable Learning Rate	traingdx	epochs, show, goal, time, min_grad, lr, mc, max_perf_inc, lr_dec, max_fail and lr_inc.

Table 3: The MATLAB Training functions and their parameters

4.3 validation of the data

The validation set is used for minimizing the overfitting. It is a part of training dataset which finds out the fine-tuned parameters for our algorithm. Moreover, the validation set is used for determining the parameters of the model. So, this data is used during training to assess how well be the performance of the network on this data may

be used to guide the training in some way (e.g., controlling the learning rate, deciding, when to stop training, choosing between several trained networks) [16].

4.4 Testing the data

The test dataset is used to evaluate the performance of the algorithm. This step has to be carried out after training, for the evaluation purposes.

After the training completes, the accuracy of the resulting neural network model's weights and biases are applied just once to the test data. The accuracy of the model on the test data gives you a very rough estimate of how accurate the model will be when presented with new, previously unseen data. So, the Test set is the valid test data and, ideally, should be used once only, after training is complete and evaluate the performance of the model in an unseen (real world) dataset [16].

The following chart summarizes all the steps:

Training set --> to fit the parameters [i.e., weights]

Validation set --> to tune the parameters [i.e., architecture]

Test set --> to assess the performance [i.e., generalization and predictive power]

5 THE IMPLEMENTATION OF NARX ALGORITHM

In this section, we have implemented time series modeling of the historical data record of stocks through a nonlinear auto- regressive exogenous model (NARX). The model relates the current values of a time series with previous values of the same series to derive a prediction model that can be matched with the current and previous values of the driving (exogenous) series. The network that has been used is dynamic in nature as it takes feedback loops (outputs of a neuron fed back to some past neuron) and taps (delay lines that are used to feed the network with previous values of inputs). The utilization of a NARX network assists in improving the accuracy of prediction in addition to can be used for real-time prediction and forecast of stock values.

The model proposed here has six sets of exogenous inputs which are

1) Opening price of the stock.

2) The highest price of the stock.

3) The lowest price of the stock.

4) Volume price of the stock.

5) DEXUSUK price of the stock.

6) DEXUSEU price of the stock.

The goal series comprises of the closing price of the stock every day. The output of the model network is calculating and predicting closing prices over the whole target series with the following value in the series. This is a curve-fitting issue which is applied using MATLAB version 2017b. A parallel NARX network is formed using "NARXNET" command which assists in improving the accuracy of the network. Moreover, we have plotted the time series model of an adapted NARX network which includes some input delays and more output delays. Hence the obtained response can be used for additional time series analysis. "Levenberg-Marquardt" (LM) algorithm is used for training of the network. It is an iterative technique which is a common alternative to the Gauss-Newton method of calculating the minimum of a function that is a summation of squares of nonlinear functions [23].

LM algorithm is a union of Gauss-Newton and Gradient Descent algorithm and therefore is more robust as it combines the speed of Newton's algorithm with the guaranteed convergence of steepest descent [23].

Gauss-Newton Method

$$\nabla t = -[\, J\, T\, (t)\, J(t)\,]\, -1\, J(t)\, e(t) \qquad (5)$$

In Levenberg-Marqardt Algorithm we use the following equation for time differentiation:

$$\nabla t = -[\, J\, T\, (t)\, J(t) + uI] \, -1\, J(t)\, e(t) \qquad (6)$$

Updating of early weights w(k) is achieved using the Sigmoid matrix and discrete error of diverse vectors

$$w(k+1) = w\{-[JT\, (t)\, J(t) + uI]-1\, J(t)\, e(t)\} \qquad (7)$$

The model of the NARX Neural network takes an input dimensionality of six, output dimensionality of one, number of hidden layer neurons are 10 and the network is trained for 1000 epochs.

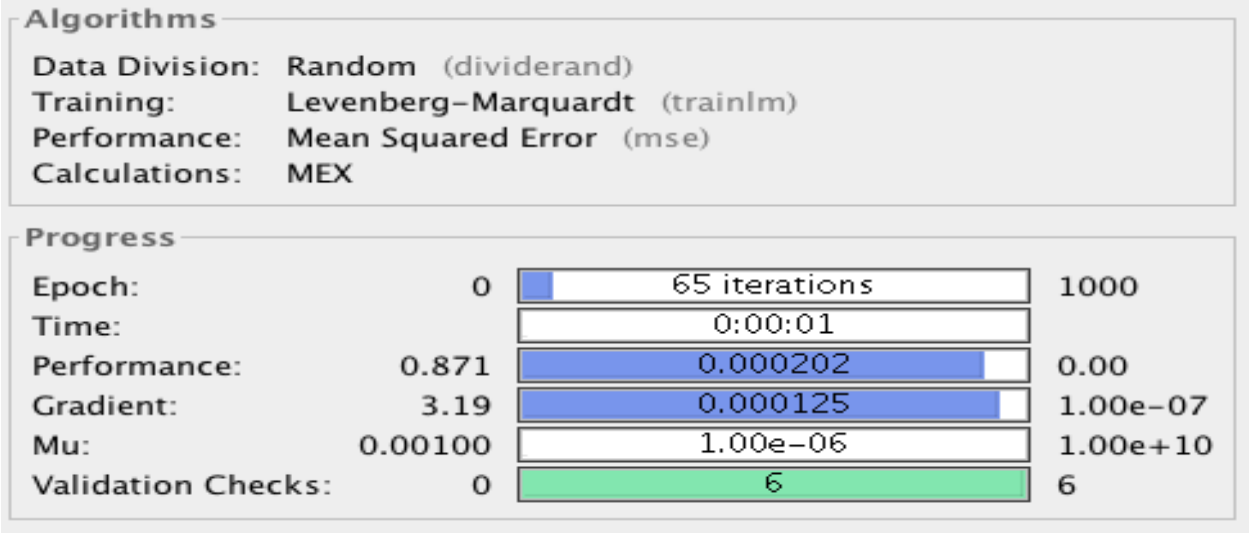


Figure 5: Algorithm and Progress

Here we used:

1- dividerand:

Divide targets into three sets using random indices

syntax: [trainInd,valInd,testInd] = dividerand(Q,trainRatio,valRatio,testRatio)

Description: splits targets into three sets: training, validation, and testing. Its parameter is the following inputs,

Q	Number of targets to divide up.
trainRatio	Ratio of vectors for training. Default = 0.7.
valRatio	Ratio of vectors for validation. Default = 0.15.
testRatio	Ratio of vectors for testing. Default = 0.15.

Table 4: The MATLAB dividerand functions and their parameters

and returns

trainInd	Training indices
valInd	Validation indices
testInd	Test indices

Table 5: The MATLAB dividerand functions returns and their parameters

2- trainlm

trainlm is a network training function that can be used to updates weight and bias values according to Levenberg-Marquardt optimization. The function rainlm is often the fastest backpropagation algorithm in the toolbox, and is highly recommended as a first-choice supervised algorithm, although it does require more memory than other algorithms.

net.trainFcn = 'trainlm' sets the network trainFcn property.

[net,tr] = train(net,...) trains the network with trainlm.

Training occurs according to trainlm training parameters, shown here with their default values:

net.trainParam.epochs	1000	Maximum number of epochs to train
net.trainParam.goal	0	Performance goal
net.trainParam.max_fail	6	Maximum validation failures
net.trainParam.min_grad	1e-7	Minimum performance gradient
net.trainParam.mu	0.001	Initial mu
net.trainParam.mu_dec	0.1	mu decrease factor
net.trainParam.mu_inc	10	mu increase factor
net.trainParam.mu_max	1e10	Maximum mu
net.trainParam.show	25	Epochs between displays (NaN for no displays)
net.trainParam.showCommandLine	false	Generate command-line output
net.trainParam.showWindow	true	Show training GUI
net.trainParam.time	inf	Maximum time to train in seconds

Table 6: The MATLAB trainlm functions and their parameters

Validation vectors are applied to stop training first if the network performance on the validation vectors fails to develop or remains the same for max_fail epochs in a row. Test vectors are applied as a further check that the network is generalizing well but do not have any impact on training. trainlm is the default training function for several network creation functions including newcf, newdtdnn, newff, and newnarx [27].

3- MSE

Mean squared normalized error performance function

Syntax: perf = mse(net,t,y,ew)

Description: mse is a network performance function. It measures the network's performance according to the mean of squared errors.

perf = mse(net,t,y,ew) takes these inputs:

net	Neural network
t	Matrix or cell array of targets
y	Matrix or cell array of outputs
ew	Error weights (optional)

Table 7: The MATLAB MSE functions and their parameters

and delivers the mean squared error.

This function has two elective parameters, which are correlated with networks whose net.trainFcn is set to this function:

1- 'regularization' can be established to any value between 0 and 1. The higher the regularization value, the more squared weights, and biases are involved in the performance estimation relative to errors. The default is 0, corresponding to no regularization [27].

2- 'normalization' can be established to 'none' (the default); 'standard', which normalizes errors between -2 and 2, corresponding to normalizing outputs and targets between -1 and 1; and 'percent', which normalizes errors between -1 and 1. This characteristic is valuable for networks with multi-element outputs. It assures that the relative accuracy of output elements with different target value ranges are handled as equally important, rather of

prioritizing the relative accuracy of the output element with the largest target value range [27].

The following table presents a summary of all implementation steps:

Neural Network	Number of layers	Hidden Neurons	Training Algorithm	Transfer Function	Max Epochs
NARX	2	10	Levenberg-Marquardt	Sigmoid	1000

Table 8: The Summary of the Implementation Steps

6 EXPERIMENTAL STUDY OF STOCK INDEX PREDICTION

6.1 Datasets and Experiment Setup

6.1.1 Data generation

The data that has been used in this research is generated from Yahoo Finance of stocks (. https://in.finance.yahoo.com/ website). We choose the Dow Jones Industrial Average Index which is the oldest and most visible market indicator in the United States. According to john & Clemens (2000) “the Dow Jones Industrial Average (DJIA) is the most quoted stock market index in the world. The changes in the index are often perceived to be representative of the American stock market” [12]. The historical dataset of the stocks we used to in our study to perform our experiment with the neural network is accessible in the form of a downloadable

different formats and styles. We have generated data of 18 years from 1st January 1999 up to 31st October 2017. The neural network is trained using this data across these years. Out of all the available samples for a particular stock, 70% forms the training data, and the remaining 30% forms the testing dataset. The output of the network gives us a prediction of the closing price of the stock for a future day.

In order to get a good result with high accuracy, we try different types of input of the dataset. also different datasets, but we get the best result when we use six inputs; Open, High Low, Volume, DEXUSUK, and DEXUSEU.

What is the ' Open, High, Low, Volume, DEXUSUK, and DEXUSEU Price'?

Open; the opening price at which a security first trades upon the opening of an exchange on a given trading day. The price of the initial trade for any stock is its every day opening price. A security's opening price is an essential marker for that day's selling movement, particularly for those involved in measuring short-term results such as day tradesmen [11].

The closing price is the concluding price at which a security is traded on a given dealing day. The closing price shows the most up-to-date valuation of security till selling begins again on the following trading day. Most financial instruments are sold after hours (although with considerably smaller volume and liquidity levels), therefore the closing price may not meet its after-hours price [11].

Low: is the lowest price at which a stock trades across the course of a selling day. Today's low is usually smaller than the opening or closing price.

Volume: is defined as, “the number of shares or contracts traded in a security or an entire market during a given period of time” [3]. Volume price is one of the common basic and useful concepts to understand when trading stocks.

Adjusted Close Price of a stock is its close price adjusted by taking into account dividends.

(DEXUSEU): U.S. / Euro Foreign Exchange Rate.

It is a concept that shows the U.S. Dollars to One Euro.

(DEXUSEU): U.S. / Euro Foreign Exchange Rate.

It is a concept that shows the U.S. Dollars to One Euro.

6.1.2 Data format and organization

The data collected was in the sort of excel/csv (comma separated values) this data was converted into the MATLAB® compatible data format namely.mat format. All the excel/csv (comma separated values) data has been converted to the MATLAB format by applying the ‘xlsread’ command of the MATLAB® environment.

6.2 Experimental Setup

6.2.1 Selection of neural network type

Neural networks can generally be classified into two parts – static and dynamic. Static networks have no feedback elements and no delays. The output is determined immediately from the current inputs. Such networks assume that the data is concurrent and no reason for time can be encoded. These networks can hence guide to instantaneous behavior.

Dynamic networks may be challenging to train but are more influential than static networks. While they have memory in the form of delays or repetitive loops, they can be trained to learn sequential or time-varying patterns. Neural networks of time-varying patterns have several applications like financial predictions, fault detection, sorting, channel equalization, speech recognition, etc. [18].

As we are working with a time series, it is important to utilize dynamic networks. Dynamic networks can be of two sorts ones with feed-forward connections and those with feedback or intermittent and recurrent networks.

At this phase, two classes of networks were examined the NARX (Nonlinear Autoregressive Neural Network) and recurrent networks. NARX networks apply taps to set up delays over the inputs and further includes the past values of the output. Recurrent networks have loops within middle layers and incorporate memory through these loops [18].

Both networks have been applied in dynamic applications. The extra time needed in training the recurrent networks is a known issue. Recurrent networks require almost 17 phases more time to finish training and simulations as compared to the NARX batch, and give a very low accuracy compared to NARX.

Dunis and Williams who have estimated recurrent network application to financial predictions have more difficulties. and there is no reason or theoretical proof that explains; outputs of layers should be looped back furthermore, the improvement in performance is low. The neuron structure of the NARX is similar to simple feedforward network [5].

As a result, our limited computational resources do not accept to train and simulate such networks. However, for applications where these gains outweigh the cost of computation, recurrent networks should be tried out [18].

According to the findings mentioned earlier and the limited computing resources available for this research, we decided that NARX networks will be used for our prediction system.

The basic NARX network is the one selected to be used for multistep predictions. When the real past values of the target are not available and only the predictions themselves are provided, they can be fed-back to the network as inputs. We are using the series-parallel version of the NARX network which is defined in section 1. Hence a series-parallel NARX dynamic network will be applied as a basis of our network.

6.2.2 The environment

The environment used for our study is MATLAB® Neural Network Toolbox. MATLAB Neural Network Toolbox has a large set of open algorithms, and related features. MATLAB also has a dedicated Distributed Computing Toolbox which can be employed to train and evaluate networks in parallel. MATLAB also gives a detailed user's manual describing different function implementations and specifications.

We are coding several functions to customize the neural network to our financial applications. Performance evaluation of our model and methodology is provided in this study [18].

6.2.3 Network architectures

The selected NARX networks have a linear input layer of neurons (default in MATLAB tool box). The Sigmoid transfer function is used for the hidden and the output layers. Tansig neurons and Sigmoid neurons have been largely applied in most of the neural network applications. A sigmoid layer has a range from 0 to 1 while the range for Tansig transfer function ranges from -1 to 1.

The Sigmoidal function has ‘S’ shaped curve in which tan hyperbolic function is applied to estimate output from the net input and the function is determined as

$$f(x) = (1/1+ \exp(-\sigma x)) \quad (8)$$

where σ is steepness parameter.

6.3 Data preprocessing

Different researchers have several ways of preprocessing the data and training a neural network for such model. They can be categorized into pattern recognition methods and functional approximation methods. In our implementation, we will apply the functional approximation method to estimate and process our dataset.

6.3.1 Function approximation

In this part, we will consider feeding the data values of series per occurrence and we use the output of the network to predict the results at every corresponding day.

The MATLAB® Neural Network Toolbox gives a comprehensive survey of algorithms appropriate for different models [9]. Results show that Levenberg-Marquardt (LM) is a good training algorithm for function approximation. On the other hand, Scaled conjugate gradient

algorithm (SCG) has good behavior for problems of large size. Consequently, we elected to use Levenberg-Marquardt (LM) algorithm for the training stage.

We will train the NARX network across all estimated values and Levenberg-Marquardt (LM) algorithm. To ensure accuracy, Mean Squared Error" is the average squared difference between outputs and targets is applied. Zero means no error" was applied to estimate the execution of the neural network models. Also, the Regression R Values estimate the correlation between outputs and targets. An R-value of 1 indicates a close relationship, 0 a random relationship.

6.3.2 Data for training, evaluation, and testing

The performance of the system was examined over different datasets of different periods of time each, thus encompassing each series. We used the most recent 18 years for all the testing (January 1999-October 2017).

Therefore, we examined the model over six-time series of data sets.

in order to predict each of these test sets, we would keep some amount of previous data for evaluation and some amount of data prior to that as the training data. Thus as we move into

consecutive test sets we should control a sliding window method to obtain the training and evaluation data.

The ratios for training, evaluation and testing can be used as (60%:20%:20%), or (60%:30%:10%) which are the most recommended and widely used ratios [9]. In our study we used (70%:15%:15%) ratios for our model. It was noted that adding more training data did not enhance results. Adding extra data to the evaluation data to prevent the training set of active new data and showed serious fall in performance.

6.3.3 Inputs for training

We use a mix of auto-regression (previous inputs), technical indicators and other currencies as inputs for our model. The order of columns in the data to be stored is as follows:

1) Opening price of the stock.

2) The highest price of the stock.

3) The lowest price of the stock.

4) Volume price of the stock.

5) DEXUSUK price of the stock.

6) DEXUSEU price of the stock.

7) Close price of time series to be predicted

8) Adjusted Close price of time series to be predicted

6.3.4 Data handling

In this section we define the functions that have been using MATLAB® to implement our system.

- Xlsread

Read Data from xls file

- mapminmax

Apply Normalization

- tonndata

Convert data to neural network suitable format

Divide data in to Train Test and validation splits

- divideFcn 'dividerand'

Divide data randomly

- divideMode = 'time'

Divide up every sample

- divideParam.trainRatio

trainlm

Define training function

Levenberg-Marquardt backpropagation.

- narxnet

Define NARX RNN Model

- preparets

Prepare data to feed the defined model

- performFcn = 'mse'

Define error measure

- train

Train defined model

- net_NARX

Test trained model

- perform

Check model performace

- Gsubtract

Check actual vs predicted difference

- save

Save model

7 RESULT AND ANALYSIS OF STOCK INDEX PREDICTION

Instead of adhering to a single architecture we had defined a range of taps and neurons and had trained and tested networks all permutation and combinations of taps and neurons within this range.

In proposed work, the NARX tool of time series prediction is used. In NARX tool, the input layer, hidden layer, delay and the activation function are declared to get the desired output. By changing the number of neuron in hidden layer and the activation function. We have to train the proposed NARX network.

The model	Best performance in training phase	Best performance in tasting phase
NARX	2.172622e-04	2.252622e-04
LR	4.450835e-01	4.560835e-01

Table 9: The Summary of the Result of Implementation NARX & LR

The NARX net model was trained with a data set of 18 years for Dow Jones stock. To analyze the working of the new neural network model for different conditions, we have divided the work into training, testing, and validating.

The best model for different layered neural networks was with ten hidden neurons, and its training algorithm was the (LM) Levenberg-Marquardt with adaptive learning rate. Its performance of predictions demonstrated an of 2.119986e-04.

The prediction graph is shown in Figure below.

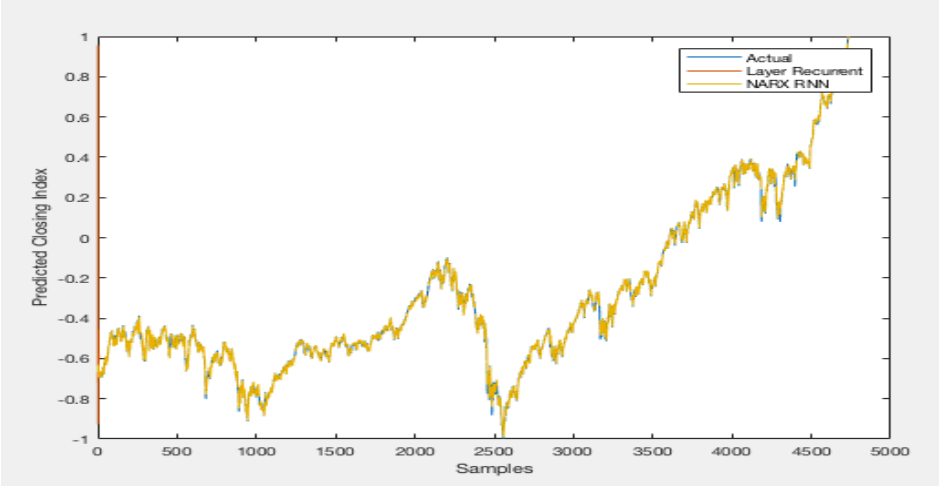


Figure 6: The Predicted Results Against Actual Output.

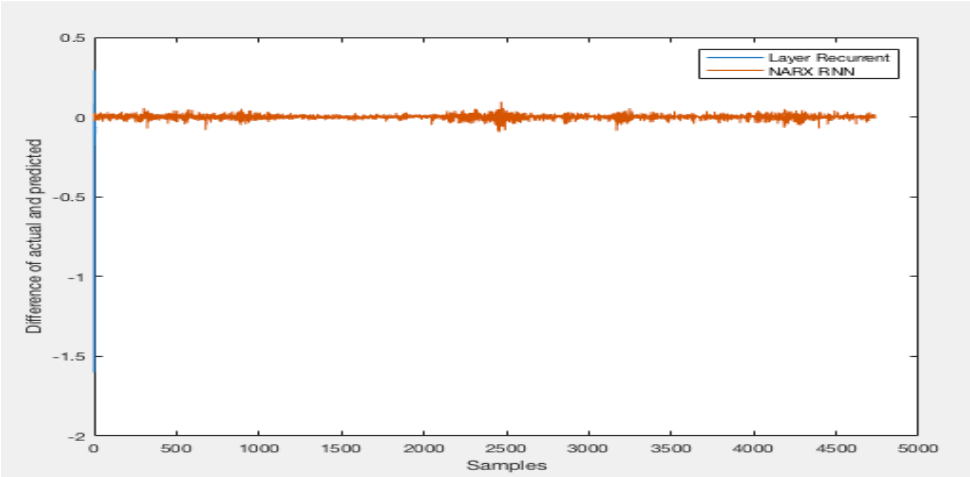


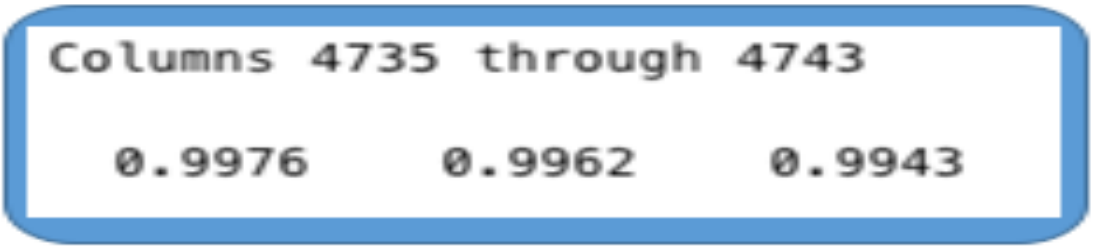
Figure 7: The Error of the Differences Between the Actual Price and The Predicted Price

The mean square error (MSE) measure is the performances criteria of the proposed model. MSE is an average squared difference between [10]. MSE is supposed to be close to zero as it means no error. Based on our experiment results after we tried different algorithms, the training algorithm with the best results was with Levenberg - Marquardt training function has reached an acceptable level of MSE after 65 iterations.

In the empirical research, NARX method has been employed by using different numbers of hidden layers and delays. Higher hidden may have smaller errors and higher R values. However, using many hidden layers decrease the importance of input variables. Even with one variable mean squared error are not very high, and R (regression) is close to 1 [10].

By using ten hidden layers and two delays, best-validated performance was at 59 Epoch

level. R-values are also showing more than 99%. All in all, the results are showing that stock index values are successfully predicted with these variables. The output given by the network is shown as:



The actual price of the stock was:

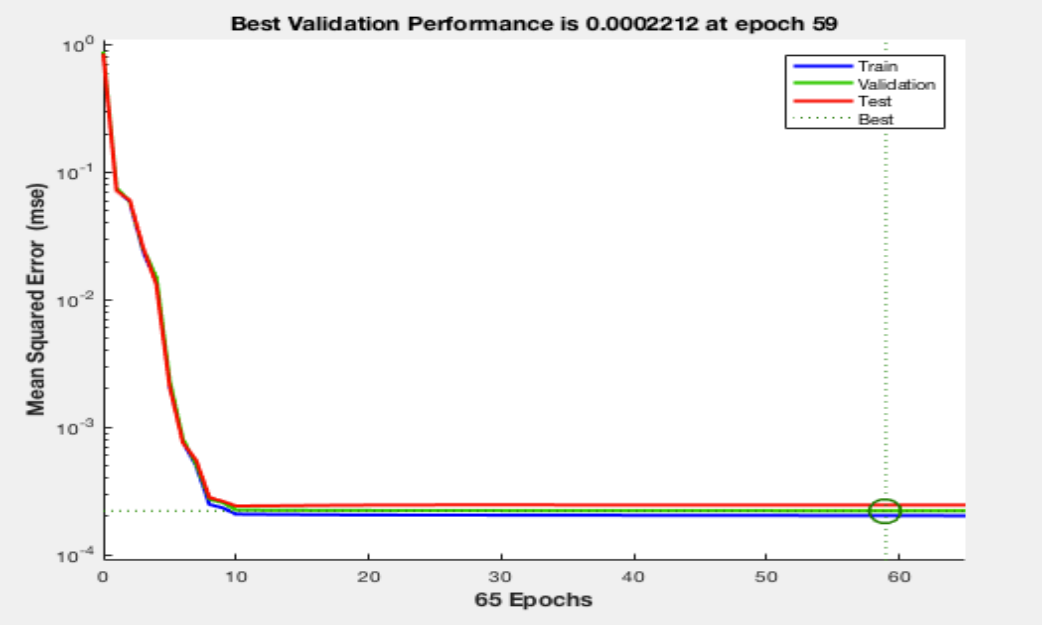


Figure 8: Graph of Performance of Training, Validation and Test Data Sets with Respect to Epochs for Stock Index Prediction Using Levenberg-Marquardt Algorithm.

The diagram shows the performance of training, validation and test data sets with respect to epochs for stock price prediction. It is seen that the best training performance was met with ten neurons in the hidden layer with Levenberg-Marquardt training algorithm, the best validation performance is 0.0002212 at 59 epochs from 65 epochs.

Moreover, the performance curve depicts how the mean square error drops rapidly as the network learns. In performance curve error is decreases means training should be stopped. The red line represents how well the network will generalize to the new data.

Figure 9: (a). Time Series Respond, (b) Error vs. Input Graph Using Levenberg-Marquardt Algorithm.

Time Series Respond figure illustrates the time series response which is displayed the inputs, targets, and errors versus time. It also indicates which time points were selected for training, testing, and validation.

So, figure 9(a) represents how the network is fit to the network. The vertical axis represents targets and outputs which are denoted with (+) and (.) respectively. Figure 9(b) shows how errors are fit to the network at the different instance of time. These curves represent the distribution of errors. Most of the errors should be as close as possible to the zero error line (line yellow).

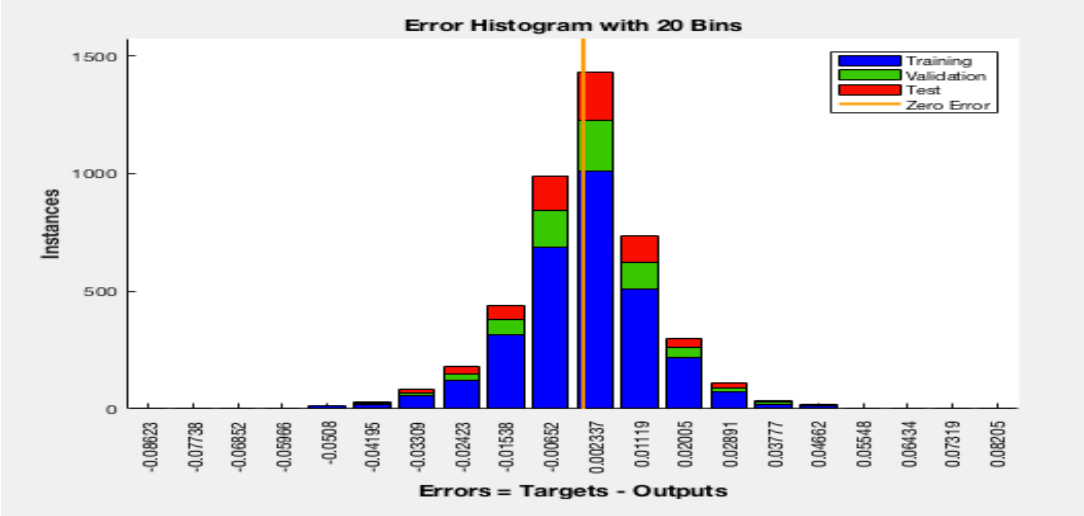


Figure 10: Neural Network Training Error Histogram using Levenberg-Marquardt Algorithm.

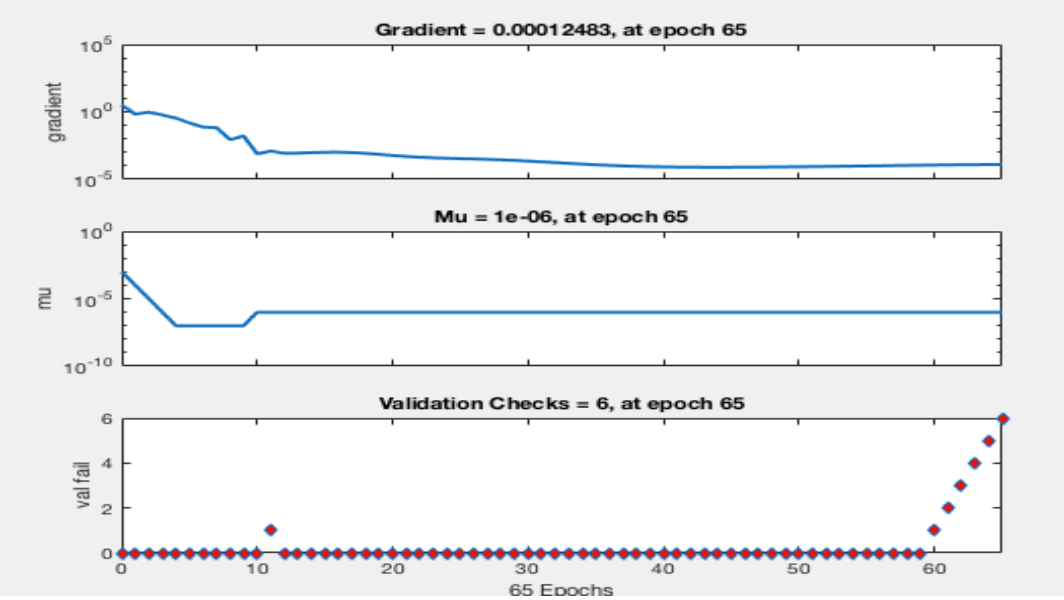


Figure 11: The Training State Value.

The figure above shows the training state value. The best validation performance is equal to 0.00012483 at epoch 59, and the training is stopped after 65 epochs. Also, the validation checks were 6 point checks.

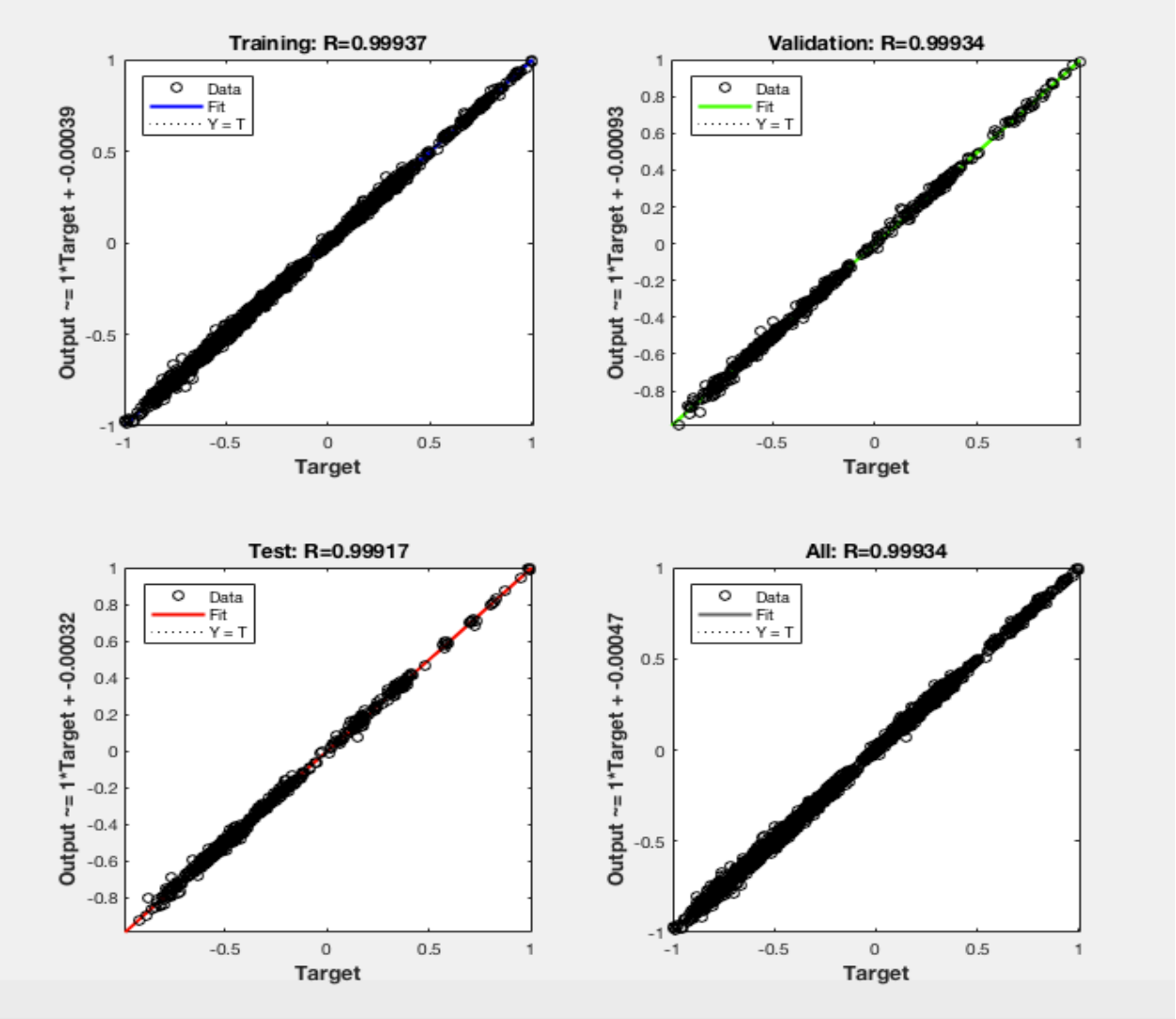


Figure 12: The Training State Value.

The Regression plots demonstrate the network outputs regard to targets for training,

validation, and test sets. For an ideal fit, the data should fall along a 45-degree line, where the network outputs are equal to the targets [19].

The figure shows the regression and explains the power of the explanation power of the model. The performance of a neural network model can be assessed by the errors on the training, validation, and test sets. R is a measure of the variation between the generated outputs and targets. When R is equal to 1, which means there is a perfect correlation between the targets and outputs. In the figures, the number is very close to 1, and this symbolizes a good fit. It can be clearly accepted that the variables can be considered as significant because R>0.99.

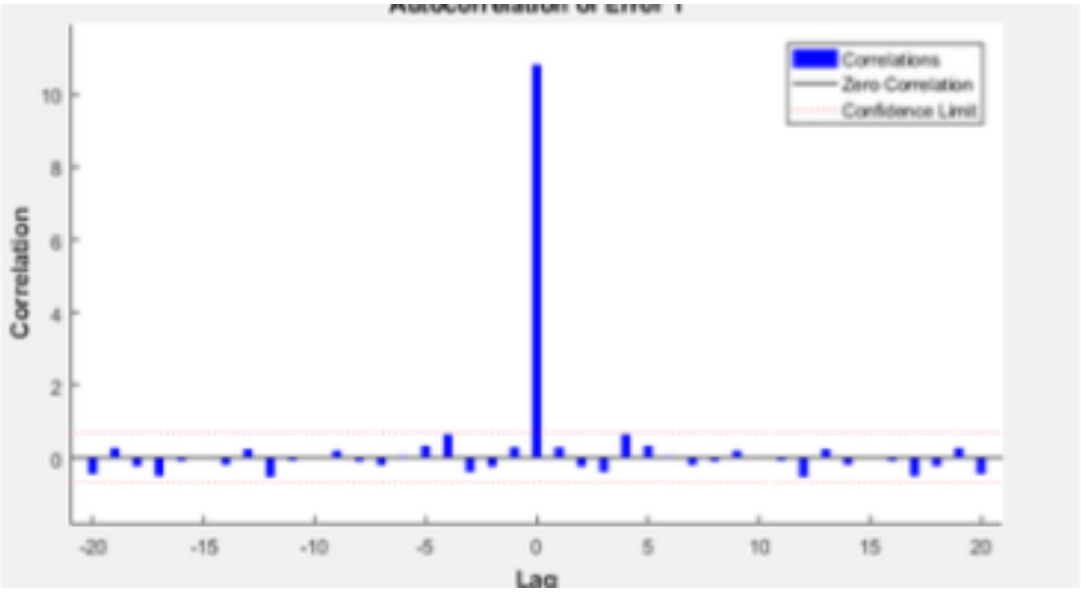


Figure 13: Autocorrelation Test

To avoid saturation and over-fitting there have been trials of different hidden layers. Because of too many neurons in the network, the prediction may be poor. Therefore, it was

essential to receive a low MSE, even with one hidden layer. Finally, the results are successful enough with ten hidden layers.

It is also visible that, increasing the number of delays leads a better result in training if higher numbers of hidden layers are used.

Autocorrelation Test shows that errors are close to being zero. It illustrates that the model does not suffer from autocorrelation.

Summary NARX RNN Model

Number of layers:	2
Hidden Neurons:	10
Transfer Function:	Sigmoid
Max Epochs:	1000
Experiment's Epochs:	65
Best Performance Epochs:	59
Performance:	2.119986e-04

Table 10: The Summary of the Best result of NARX

8 CONCLUSIONS

Exact forecasting of future stock price plays a vital role in global economy. Although Stock market prediction has been implemented using neural network algorithms, the NARX neural network model results in better accuracy. The capability of the NARX neural network to incorporate time delays outcomes in the better computation of stock price result based on time

series feedback. Implement NARX neural network in the open loop with input delay designs and accordingly is the optimum configuration to confirm the better performance of the model.

Since the accuracy in the prediction of future price is the most critical aspect, the proposed model provides a high accuracy by 98% of prediction for stable and moderately stable stocks. The NARX model is trained by using SIX input values - the opening, the closing, the highest, the lowest price, the volume, DEXUSUK price, and DEXUSEU price of the stock for the day and more features. Moreover, the target values are also fed to the system by way of it is a supervised learning model. Besides, in this model, we have made use of an open loop with the used of the input delays for making predictions and the accuracy of the situation is determined to analyze the functioning of a NARX neural network and to determine the optimum configuration.

REFERENCES

[1] A. Thakur, A. Tiwari, S. Kumar, A. Jain and J. Singh, "NARX Based Forecasting of Petrol Prices", in 5th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO) (Trends and Future Directions),, 2016, pp. 610 - 614.

[2] B. Graham, D. Dodd, and W. Buffett, (2009). Security analysis. New York: McGraw-Hill.

[3] B. Reinkensmeyer, "StockTrader.com" (n.d.). Jun 16th, 2011 Retrieved December 27, 2017, from https://www.stocktrader.com/

[4] B. U. Devi, D. Sundar, and P. Alli, "An optimized approach to predict the stock market behavior and investment decision making using benchmark algorithms for Naïve investors," 2013 IEEE International Conference on Computational Intelligence and Computing Research, 2013.

[5] C. Dunis, M. Williams, "Modeling and Trading the EUR/USD Exchange Rate: Do Neural Network Models Perform Better?", Liverpool Business School and CIBEF, 2002

[6] C. Wang, "Time series neural network systems in stock index forecasting", Computer Modelling & New Technologies, vol. 19, no. 1, pp. 57 - 61, 2015.

[7] E. Diaconescu, "The use of NARX Neural Networks to predict Chaotic Time Series", *WSEAS Transactions on Computer Research*, vol. 3, no. 3, pp. 182 - 192, 2017.

[8] G. Rajput and B. Kaulwar, "Artificial neural network (ann) based prediction of stock closing price in nse of india", 2017, pp. 19 - 27.

[9] H. Demuth, M. Beale, M. Hagan,"MATLAB Neural Network Toolbox 5, Users Guide", 2007

[10] H. Ercan, "Baltic Stock Market Prediction by Using NARX", 2017, pp. 464 - 467.

[11] Investopedia. (n.d.). Retrieved January 04, 2018, from https://investopedia.adblade.com

[12] J. Shoven, & C. Sialm, "The Dow Jones Industrial Average". The Journal of Wealth Management, 3(3), 9-18. doi:10.3905/jwm.2000.320332

[13] L. Crnkovic-Friis and M. Erlandson, "Geology Driven EUR Prediction Using Deep Learning," SPE Annual Technical Conference and Exhibition, 2015.

[14] M. Dunne, "Stock Market Prediction" (Unpublished thesis). University College Cork. Retrieved December 26, 2017.

[15] M. R. Vargas, Beatriz S. L. P. De Lima, and A. G. Evsukoff, "Deep learning for stock market prediction from financial news articles," 2017 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA), 2017.

[16] McCaffrey, "Neural Network Train-Validate-Test Stopping" (2015, May 13).

[17] N. Shahbazi, M. Memarzadeh, and J. Gryz, “Forex Market Prediction Using NARX Neural Network with Bagging,” MATEC Web of Conferences, vol. 68, p. 19001, 2016.

[18] P. Soman, “An adaptive NARX neural network approach for financial time series prediction” (Unpublished thesis). The State University of New Jersey (2008).

[19] Q. Al-Shayea, "Neural Networks to Predict Stock Market Price", in World Congress on Engineering and Computer Science, San Fransisco, 2017, pp. 1 - 7.

[20] R. Cavalcante, R. Brasileiro, V. Souza, J. Nobrega and A. Oliveira, "Computational Intelligence and Financial Markets: A Survey and Future Directions", Expert Systems With Applications, vol. 55, no. 1, pp. 194 - 211, 2016.

[21] R. Singh and S. Srivastava, “Stock prediction using deep learning,” Multimedia Tools and Applications, vol. 76, no. 18, pp. 18569–18584, 2016.

[22] S. Chaigusin, C. Chirathamjaree, and J. Clayden, “The Use of Neural Networks in the Prediction of the Stock Exchange of Thailand (SET) Index,” 2008 International Conference on Computational Intelligence for Modelling Control & Automation, 2008.

[23] S. Labde, S. Patel and M. Shukla, "Time Series Regression Model for Prediction of Closing Values of the Stock using an Adaptive NARX Neural Network", International Journal of Computer Applications, vol. 158, no. 10, pp. 29 - 35, 2017.

[24] T. Gao, X. Li, Y. Chai, and Y. Tang, "Deep learning with stock indicators and two-dimensional principal component analysis for closing price prediction system," 2016 7th IEEE International Conference on Software Engineering and Service Science (ICSESS), 2016.

[25] "Training an Artificial Neural Network – Intro" (2012, August 02).

[26] W. Chen, Y. Zhang, C. K. Yeo, C. T. Lau, and B. S. Lee, "Stock market prediction using neural network through news on online social networks," 2017 International Smart Cities Conference (ISC2), 2017.

[27] www.mathworks.com/help/nnet/ref/trainlm.html.

[28] Y. Yetis, H. Kaplan, and M. Jamshidi, "Stock market prediction by using artificial neural network," 2014 World Automation Congress (WAC), 2014.